2013

Modificación de brazo robótico para su automatización

CON ARDUINO

DANIEL LOZANO, ANGEL COTA

UNIVERSIDAD DE NAVOJOA | INGENIERIA EN SISTEMAS COMPUTACIONALES

Contenido

Resumen....3
Introducción....3
Planteamiento del problema....3
Objetivos del Proyecto....4
Justificación del Proyecto....4
Materiales....4
Procedimiento....4
Armado del brazo....4
Circuito....5
Protoboard....5
L293D....6
Cables....7
Finalizando Circuito....7
Código de prueba....8
ARDUINO....10
Preguntas y Respuestas....10
? Que es Arduino?....10
¿En qué lenguajes se programa?....11
¿Dónde descargo el Software y Drivers de mi Arduino?....11
¿Qué es processing?....11
¿CODIGO DEL ARDUINO?....12
Controlar el Arduino con TECLADO....14

Resumen

Este proyecto consiste en la manipulación por medios remotos y de manera automatizada, el control de un brazo robótico ejecutando tareas autónomas, pues al programar la ruta tiene la capacidad de repetir las rutas automáticamente.

Palabras claves: Medios remotos, automatizada, brazo robótico, autónomas.

Introducción.

La robótica industrial puede definirse como el estudio, diseño y uso de robots para la ejecución de procesos industriales. La definición que maneja, el estándar ISO (ISO 8373:1994, Robots industriales manipuladores – Vocabulario) define un robot industrial como un manipulador programable en tres o más ejes multipropósito, controlado automáticamente y reprogramable.

Este proyecto tiene como finalidad la manipulación remota y de forma automatizada el control de un brazo robótico. Para simular el trabajo de un brazo humano imitando sus funciones.

Planteamiento del problema.

El brazo robótico se basa en la utilización de servomotores de transmisión directa, y actuadores conectados a un panel de control (interfaz de usuario) donde se controlará y programará la tarea a desarrollar. Estos a su vez realizarán los movimientos vertical y horizontal con el fin de cumplir la tarea establecida por el usuario.

Objetivos del Proyecto

-Utilizar un código para programar la interfaz de usuario del prototipo

-Hacer uso de diferentes medios de control remoto

-Realizar un análisis comparativo del desempeño del brazo robot con respecto a uno comercial

- Elaborar el manual de usuario

Justificación del Proyecto

En la actualidad utilizar sistemas automatizados tiene como resultado una tarea atractiva e indispensable dentro de los procesos industriales. Es por eso que se está dirigiendo este proyecto a la manipulación de un brazo robótica para que realice los trabajos principalmente para el uso industrial.

Materiales

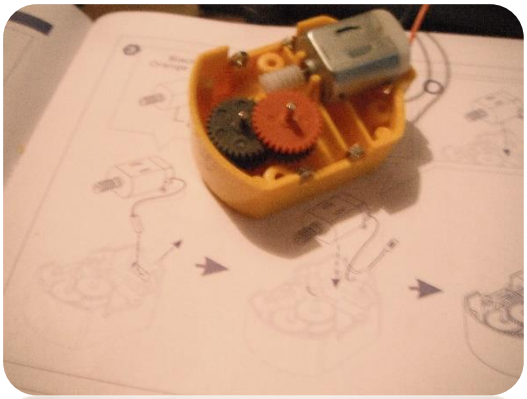
#	Nombre	Precio US	Precio MX	Link	Total
3	L293D Driver de motor	$ 6.99	$ 97.86	Link	$ 293.58
1	OWI Robotic Arm Edge	$ 47.99	$ 671.86	Link	$ 671.86
1	Arduino UNO R3	$ 23.00	$ 322.00	Link	$ 322.00
1	ProtoBoard	$ 8.00	$ 112.00	Link	$ 112.00
				TOTAL	$ 1,399.44

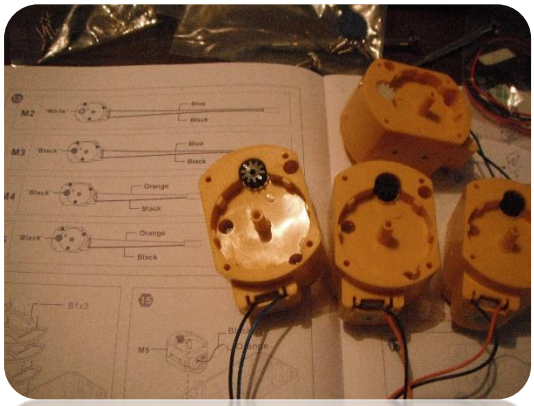
Procedimiento

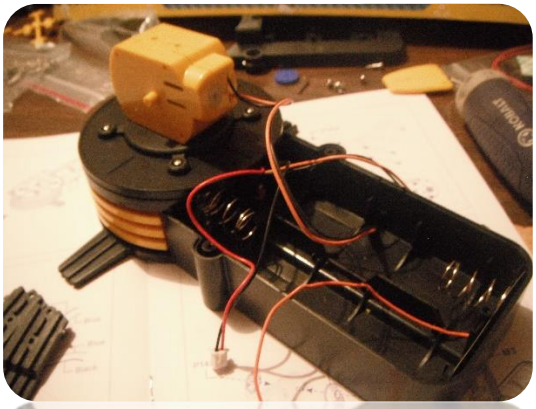
Armado del brazo

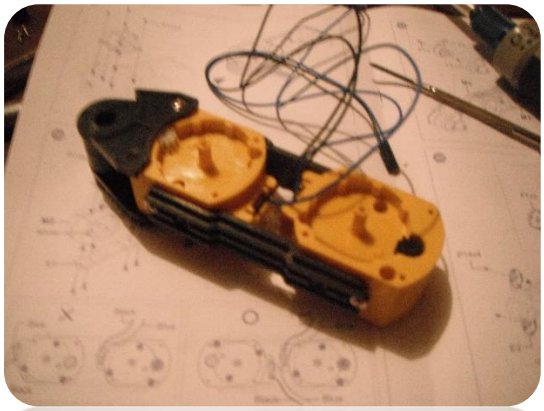
Primero debemos de comenzar armando el brazo con mucho cuidado, siguiendo claramente los instructivos que vienen en la caja, se comenzaran

armando los motores, hasta llegar a todo el brazo, es recomendable guiarte bien por los instructivos para identificar los diferentes tornillos.





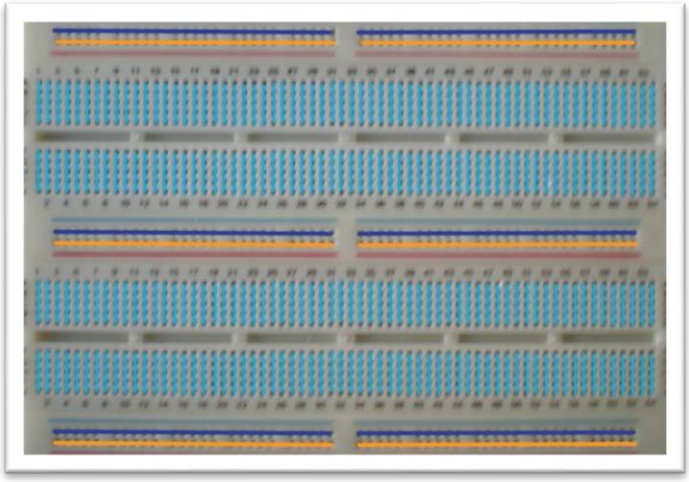




Circuito

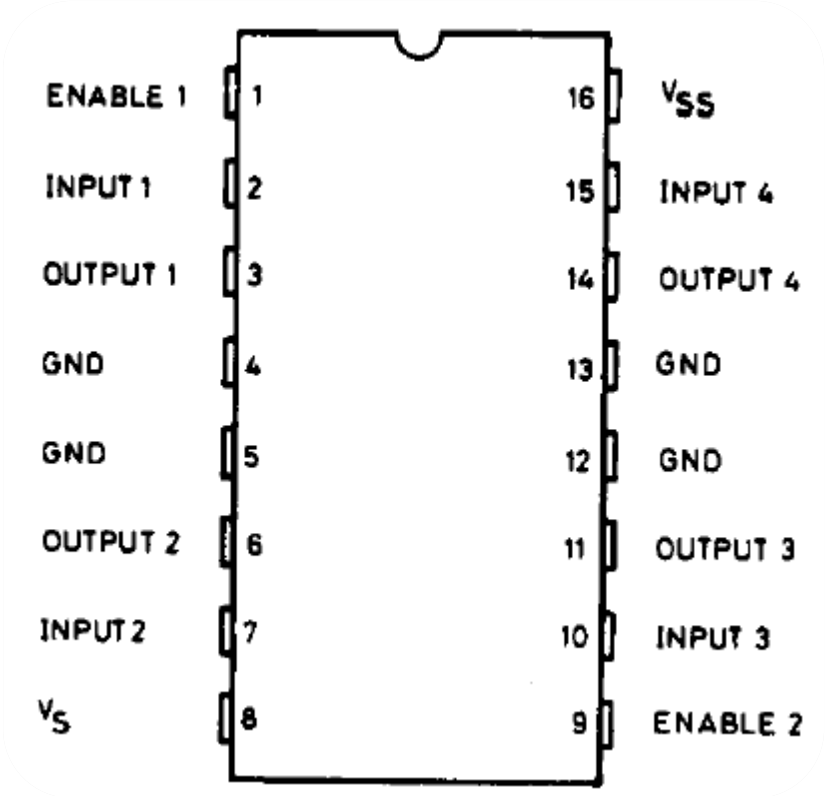
Protoboard

Es importante entender cómo funciona el protoboard antes de continuar con el circuito, los rieles de la orilla son para meter corriente + y – están unidos de la siguiente manera.



Como podemos ver donde tenemos los huecos irán los chips (L293D) ya que es muy importante que ninguna de los 16 patitas peguen en ningún lugar.

L293D



En la foto de arriba explica cómo se conectara las 16 patitas del controlador, para hacerlo funcionar necesita una alimentación de voltaje y tierra también

le falta 3 entradas para cada motor, y 2 salidas donde sacara los voltajes de los motores. Por ejemplo:

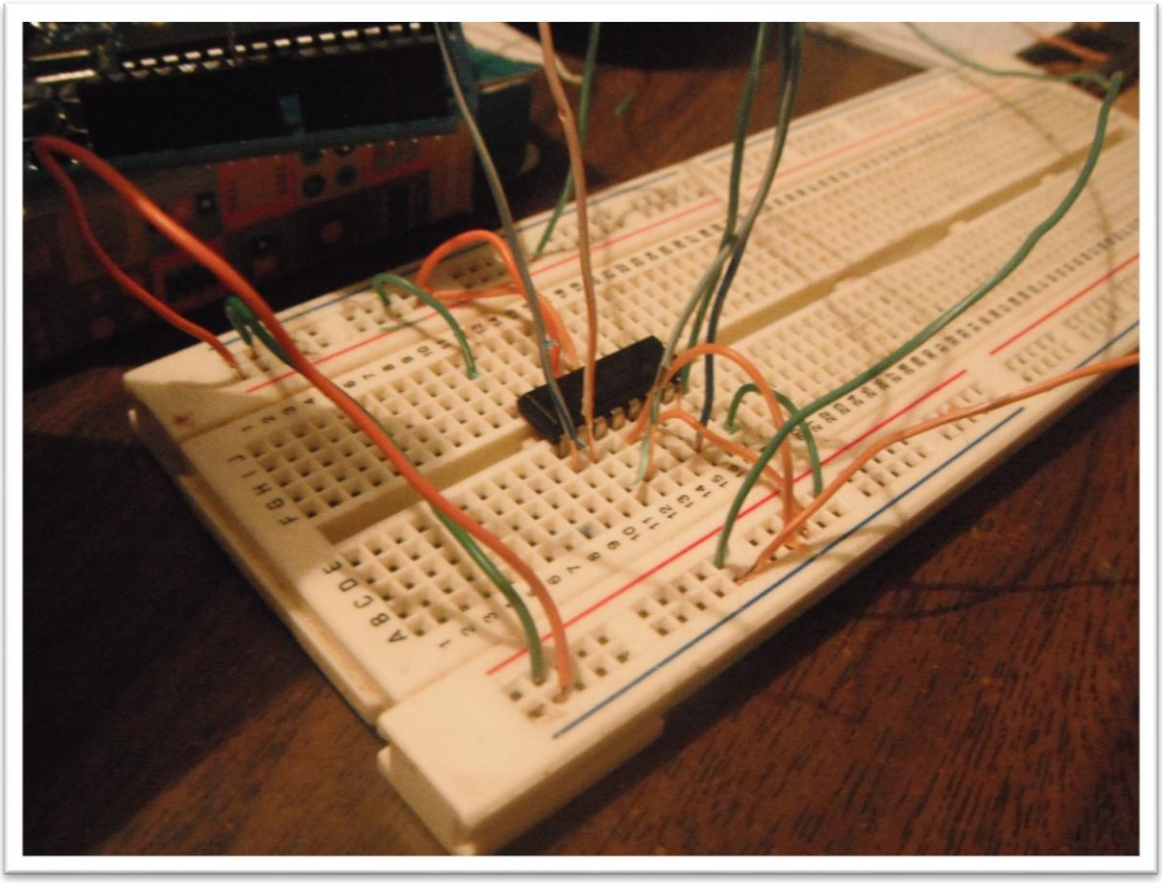
- Las patitas 4, 5, 12, 13 van conectadas a tierra.
- Las patitas 8 y 16 van conectadas a un voltaje de hasta 30 Volts.
- En el lado izquierdo van 3 entras que serían las patitas 1, 2 y 7.
- Y por último las patitas 3 y 6 es donde irían contados los motores.

Cables

Para los cables, se pueden usar Jumpers, o en mi caso use cable de Red Categoría 5. Se necesitaran bastantes cables así que es recomendable empezar a cortar y sacarles más o menos, 1 cm de punta de cada lado.

Finalizando Circuito

Comencemos con un solo chip hasta que funcione para evitar quemarlos. Si algo nos sale más quemaremos solo uno y no los tres completos.

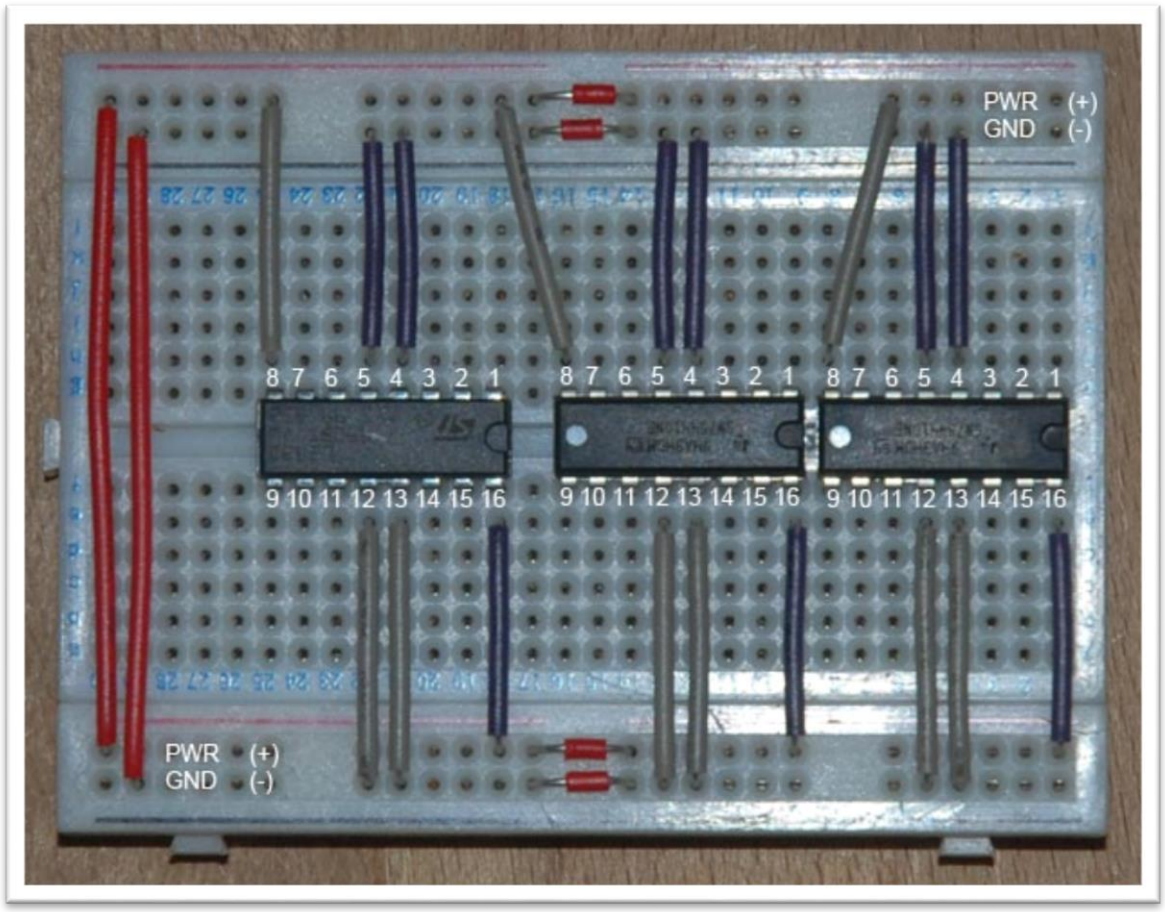


Para conectar los motores los cablearemos de la siguiente manera.

- chip 1, pin 1 a Arduino pin 14 (Analog pin o)

- chip 1, pin 2 a Arduino pin 15 (Analog pin 1)
- chip 1, pin 7 a Arduino pin 16 (Analog pin 2)

Es más fácil seguir el siguiente diagrama:

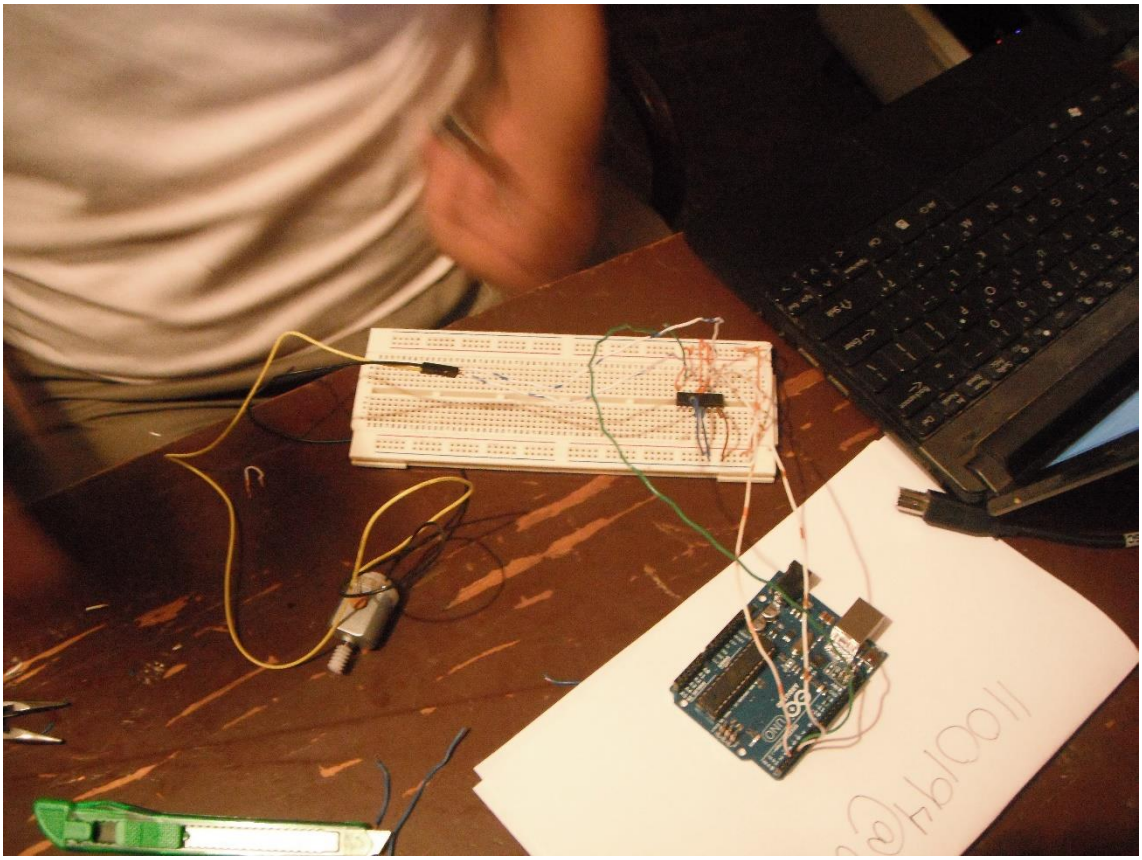


Código de prueba

Para probar los chips se puede usar el siguiente código en el Arduino.

```
int baseMotorEnablePin = 2;
int baseMotorPin1 = 3;
int baseMotorPin2 = 4;
int shoulderMotorEnablePin = 14;
int shoulderMotorPin1 = 15;
int shoulderMotorPin2 = 16;
int elbowMotorEnablePin = 8;
int elbowMotorPin1 = 9;
int elbowMotorPin2 = 10;
int wristMotorEnablePin = 5;
int wristMotorPin1 = 6;
int wristMotorPin2 = 7;
int handMotorEnablePin = 11
int handMotorPin1 = 17;
```

```
int handMotorPin2 = 18;

void setup() {


  pinMode(baseMotorPin1, OUTPUT);
  pinMode(baseMotorPin2, OUTPUT);
  pinMode(baseMotorEnablePin, OUTPUT);
  digitalWrite(baseMotorEnablePin, HIGH);
  pinMode(shoulderMotorPin1, OUTPUT);
  pinMode(shoulderMotorPin2, OUTPUT);
  pinMode(shoulderMotorEnablePin, OUTPUT);
  digitalWrite(shoulderMotorEnablePin, HIGH);
  pinMode(elbowMotorPin1, OUTPUT);
  pinMode(elbowMotorPin2, OUTPUT);
  pinMode(elbowMotorEnablePin, OUTPUT);
  digitalWrite(elbowMotorEnablePin, HIGH);
  pinMode(wristMotorPin1, OUTPUT);
  pinMode(wristMotorPin2, OUTPUT);
  pinMode(wristMotorEnablePin, OUTPUT);
  digitalWrite(wristMotorEnablePin, HIGH);

}

void loop() {
    /*
    // Para hacer que cualquier motor se mueva solo hay
que cambiar un LOW por un HIGH.

    digitalWrite(baseMotorPin1, LOW);
    digitalWrite(baseMotorPin2, LOW);
    digitalWrite(shoulderMotorPin1, LOW);
    digitalWrite(shoulderMotorPin2, LOW);
    digitalWrite(elbowMotorPin1, LOW);
    digitalWrite(elbowMotorPin2, LOW);
    digitalWrite(wristMotorPin1, LOW);
    digitalWrite(wristMotorPin2, LOW);
    */

}
```



Para probarlo basta solo con un motor dc de unos 3 a 5 voltios.

ARDUINO

Preguntas y Respuestas

? Que es Arduino?

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles muy fáciles de usar, debido a que el IDE con el que trabaja es fácil de aprender a utilizar, y el lenguaje de programación con el que trabaja es simple, pues se creó para artistas, diseñadores, aficionados y cualquier interesado en crear entornos u objetos interactivos. Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces, motores y otros actuadores. El micro controlador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un ordenador, si bien tienen la posibilidad de hacerlo y comunicar con diferentes tipos de software

¿En qué lenguajes se programa?

La plataforma Arduino se programa mediante el uso de un lenguaje propio basado en el popular lenguaje de programación de alto nivel Processing. Sin embargo, es posible utilizar otros lenguajes de programación y aplicaciones populares en Arduino.10 Algunos ejemplos son:

- 3DVIA Virtools
- Adobe Director
- BlitzMax
- C
- C++
- C#
- Cocoa/Objective-C
- Flash
- Gambas
- Isadora
- Java
- Liberlab
- Mathematica
- Matlab
- MaxMSP
- Minibloq
- Perl
- Php
- Physical Etoys
- Processing
- Pure Data
- Python
- Ruby
- Scratch for Arduino
- Squeak
- SuperCollider
- VBScript
- Visual Basic .NET
- VVVV

Esto es posible debido a que Arduino se comunica mediante la transmisión de datos en formato serie que es algo que la mayoría de los lenguajes anteriormente citados soportan. Para los que no soportan el formato serie de forma nativa, es posible utilizar software intermediario que traduzca los mensajes enviados por ambas partes para permitir una comunicación fluida. Es bastante interesante tener la posibilidad de interactuar Arduino mediante esta gran variedad de sistemas y lenguajes puesto que dependiendo de cuales sean las necesidades del problema que vamos a resolver podremos aprovecharnos de la gran compatibilidad de comunicación que ofrece.

¿Dónde descargo el Software y Drivers de mi Arduino?

Siempre se pondrá obtener la versión más actual de Arduino desde el siguiente link http://www.arduino.cc/en/Main/Software.

¿Qué es processing?

Processing es un lenguaje de programación y entorno de desarrollo integrado de código abierto basado en Java, de fácil utilización, y que sirve como medio para la enseñanza y producción de proyectos multimedia e interactivos de diseño digital. Fue iniciado por Ben Fry y Casey Reas a partir de reflexiones

en el Aesthetics and Computation Group del MIT Media Lab dirigido por John Maeda.

Su última versión se puede descargar desde https://processing.org/download/,

¿CODIGO DEL ARDUINO?

En el Arduino programaremos diferentes funciones dándole a conocer que es lo que tiene que hacer cuando recibe diferentes paquetes por el puerto serial. Por ejemplo si recibe una "a" que mande corriente por el pin 12 y al recibir una "t" que la detenga.

```
// Definir pins de salida
// 14-18 son pines analogos 0-4
int baseMotorEnablePin = 2;
int baseMotorPin1 = 3;
int baseMotorPin2 = 4;
int shoulderMotorEnablePin = 14;
int shoulderMotorPin1 = 15;
int shoulderMotorPin2 = 16;
int elbowMotorEnablePin = 8;
int elbowMotorPin1 = 9;
int elbowMotorPin2 = 10;
int wristMotorEnablePin = 5;
int wristMotorPin1 = 6;
int wristMotorPin2 = 7;
int handMotorEnablePin = 11;
int handMotorPin1 = 17;
int handMotorPin2 = 18;
// definir la variable para almacenar el bit mandado x serial
int incomingByte;

void setup() {
  // definir los pins como salida:
  pinMode(baseMotorPin1, OUTPUT);
  pinMode(baseMotorPin2, OUTPUT);
  pinMode(baseMotorEnablePin, OUTPUT);
  digitalWrite(baseMotorEnablePin, HIGH);
  pinMode(shoulderMotorPin1, OUTPUT);
  pinMode(shoulderMotorPin2, OUTPUT);
  pinMode(shoulderMotorEnablePin, OUTPUT);
  digitalWrite(shoulderMotorEnablePin, HIGH);
  pinMode(elbowMotorPin1, OUTPUT);
  pinMode(elbowMotorPin2, OUTPUT);
  pinMode(elbowMotorEnablePin, OUTPUT);
  digitalWrite(elbowMotorEnablePin, HIGH);
  pinMode(wristMotorPin1, OUTPUT);
```

```
  pinMode(wristMotorPin2, OUTPUT);
  pinMode(wristMotorEnablePin, OUTPUT);
  digitalWrite(wristMotorEnablePin, HIGH);
  pinMode(handMotorPin1, OUTPUT);
  pinMode(handMotorPin2, OUTPUT);
  pinMode(handMotorEnablePin, OUTPUT);
  digitalWrite(handMotorEnablePin, HIGH);
  // comenzar a mandar a una fecuencia de 9600
  Serial.begin(9600);
}

void loop() {
  // chekar si existe algo en el buffer serial
  if (Serial.available() > 0) {
    // leer byte y almancenarlo en varible
    // mandar byte en su valor ascii
    incomingByte = Serial.read();
    // cada letra gira los motores
    if (incomingByte == 'Q') {
    digitalWrite(baseMotorPin1, LOW);
    digitalWrite(baseMotorPin2, HIGH);
    }
    if (incomingByte == 'W') {
    digitalWrite(baseMotorPin1, HIGH);
    digitalWrite(baseMotorPin2, LOW);
    }
    if (incomingByte == 'E') {
    digitalWrite(shoulderMotorPin1, LOW);
    digitalWrite(shoulderMotorPin2, HIGH);
    }
    if (incomingByte == 'R') {
    digitalWrite(shoulderMotorPin1, HIGH);
    digitalWrite(shoulderMotorPin2, LOW);
    }
    if (incomingByte == 'A') {
    digitalWrite(elbowMotorPin1, LOW);
    digitalWrite(elbowMotorPin2, HIGH);
    }
    if (incomingByte == 'S') {
    digitalWrite(elbowMotorPin1, HIGH);
    digitalWrite(elbowMotorPin2, LOW);
    }
    if (incomingByte == 'D') {
    digitalWrite(wristMotorPin1, LOW);
    digitalWrite(wristMotorPin2, HIGH);
    }
    if (incomingByte == 'F') {
    digitalWrite(wristMotorPin1, HIGH);
    digitalWrite(wristMotorPin2, LOW);
    }
    if (incomingByte == 'Z') {
    digitalWrite(handMotorPin1, LOW);
    digitalWrite(handMotorPin2, HIGH);
```

```
    }
    if (incomingByte == 'X') {
    digitalWrite(handMotorPin1, HIGH);
    digitalWrite(handMotorPin2, LOW);
    }
    // Si entra una 0 apagar todos los motores.
    if (incomingByte == 'O') {
    digitalWrite(baseMotorPin1, LOW);
    digitalWrite(baseMotorPin2, LOW);
    digitalWrite(shoulderMotorPin1, LOW);
    digitalWrite(shoulderMotorPin2, LOW);
    digitalWrite(elbowMotorPin1, LOW);
    digitalWrite(elbowMotorPin2, LOW);
    digitalWrite(wristMotorPin1, LOW);
    digitalWrite(wristMotorPin2, LOW);
    digitalWrite(handMotorPin1, LOW);
    digitalWrite(handMotorPin2, LOW);
    }
  }
}
```

Este condigo se subirá usando el Software del Arduino al Arduino, básicamente lo que dice es que espera que le llegue algún carácter y una vez que recibe uno realiza una acción.

Controlar el Arduino con TECLADO

Para controlar el Arduino usaremos el programa prossesing para mandarle las caracteres, por medio del teclado. Básicamente lo que aremos es crear listeners que esperen a que se presione un tecla y una vez presionada mande la tecla que se allá presionado.

```
import processing.serial.*;
Serial port;
PFont myFont;

void setup() {

  println(Serial.list());
  size(145, 145);

  port = new Serial(this, Serial.list()[0], 9600);
}

void draw()
{
  background(0);
  noStroke();
  fill(150);
  // draw each box/ button with a label above each
  text("CONTROLANDO!", 5, 5, 200, 75);
```

```
  if (keyPressed)
  {
    switch (key) {
    case 'q':
    case 'Q':
      port.write('Q');
      break;
    case 'w':
    case 'W':
      port.write('W');
      break;
    case 'e':
    case 'E':
      port.write('E');
      break;
    case 'r':
    case 'R':
      port.write('R');
      break;
    case 'a':
    case 'A':
      port.write('A');
      break;
    case 's':
    case 'S':
      port.write('S');
      break;
    case 'd':
    case 'D':
      port.write('D');
      break;
    case 'f':
    case 'F':
      port.write('F');
      break;
    case 'z':
    case 'Z':
      port.write('Z');
      break;
    case 'x':
    case 'X':
      port.write('X');
      break;
    case 'o':
    case 'O':
      port.write('O');
      break;
    case 't':
    case 'T':
      port.write('T');
      break;
    default:
      break;

    }
  }
  else {
    port.write('O');
  }
}
```